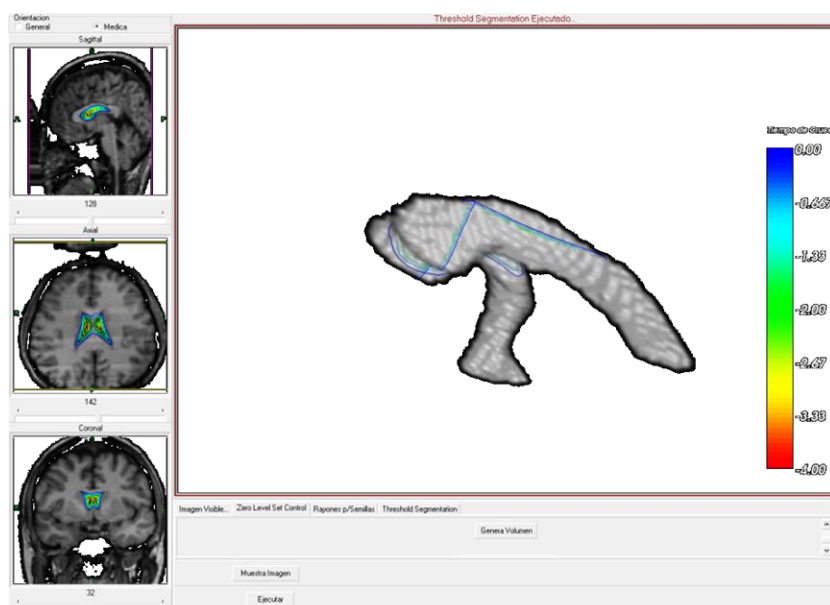


Abril 2010



Como Compilar Proyectos con C++ Borland Builder, ITK y VTK

CONTENIDO

<i>CARACTERÍSTICAS DEL SISTEMA</i>	4
<i>CARACTERÍSTICAS</i>	4
<i>INSTALAR C++ BUILDER</i>	4
<i>MOTIVACIÓN</i>	4
<i>APLICACIONES OPCIONALES</i>	6
<i>INSTALAR VISIBROKER (OPCIONAL)</i>	6
<i>INSTALAR INTERBASE CLIENT (OPCIONAL)</i>	8
<i>PROBANDO LA INSTALACIÓN DE BORLAND® C++ BUILDER 5.3</i>	9
<i>INSTALAR EL COMPILADOR BORLAND C++ 5.5</i>	10
<i>CMAKE, ITK Y VTK</i>	11
<i>CMAKE HTTP://WWW.CMAKE.ORG/</i>	11
<i>ITK HTTP://WWW.ITK.ORG/</i>	11
<i>VTK HTTP://WWW.VTK.ORG</i>	11
<i>COMPILER ISSUES, BORLAND BCC 5.5</i>	11
<i>INSTALAR CMAKE</i>	12
<i>COMPILAR ITK</i>	12
<i>DESCOMPIMIR [ITK_CODE]</i>	12
<i>GENERAR BINARIOS ITK [ITK_BINARY]</i>	12
<i>GENERAR INSTALADORES [ITK_INSTALL]</i>	14
<i>COMPILAR ITK</i>	14
<i>COMPILAR VTK</i>	14
<i>DESCOMPIMIR [VTK_CODE]</i>	14
<i>GENERAR BINARIOS VTK [VTK_BINARY]</i>	14
<i>GENERAR INSTALADORES [VTK_INSTALL]</i>	16
<i>COMPILAR VTK</i>	16
<i>BORLAND C++ BUILDER® & VTK</i>	18

BORLAND C++ BUILDER

<i>INSTALAR PAQUETE VTKBORLANDRENDERWINDOW EN BORLAND.....</i>	<i>18</i>
<i>DEMO VTKBORLAND.....</i>	<i>21</i>
<i>BORLAND C++ BUILDER® & ITK.....</i>	<i>22</i>
<i>BORLAND C++ BUILDER®, ITK & VTK.....</i>	<i>24</i>
<i>ERRORES</i>	<i>28</i>
<i>MAILING LISTS.....</i>	<i>29</i>
<i>AL FINAL... ..</i>	<i>29</i>

CARACTERÍSTICAS DEL SISTEMA

MOTIVACIÓN

Este texto es generado con la finalidad de facilitar la configuración de las herramientas que permiten desarrollar aplicaciones de visualización y procesamiento de imágenes 3D.

Dichas herramientas no son únicas, fueron las utilizadas para generar MARVIN y MARVIN_LS, dos aplicaciones generadas en CIMAT A. C.

Borland C++ Builder (BCB) Ofrece la capacidad de desarrollo de aplicaciones GUI.

Insight ToolKit (ITK) Herramienta de procesamiento de datos multi-dimensionales, en particular 3D.

Visualization ToolKit (VTK) Herramientas de visualización.

Aunque el manual esta generado para ser seguido paso a paso se pide al lector conocimientos básicos/intermedios de programación C++ orientada a objetos (de preferencia en BCB), así como conocimiento básico en procesamiento de imágenes, y visualización de datos.

En este apartado se detallan las características del sistema de cómputo en el que se realizó una instalación de prueba, la cual permitió generar este *manual* de instalación para compilar proyectos Con Borland C++ Builder, ITK y VTK . Para ver características mínimas de instalación para cada elemento, refiéranse a las fuentes originales de las librerías ITK, VTK, Borland C++ Builder y Compilador Borland C++ 5.5.

Características

Intel® Pentium® 4, 3.0 GHz

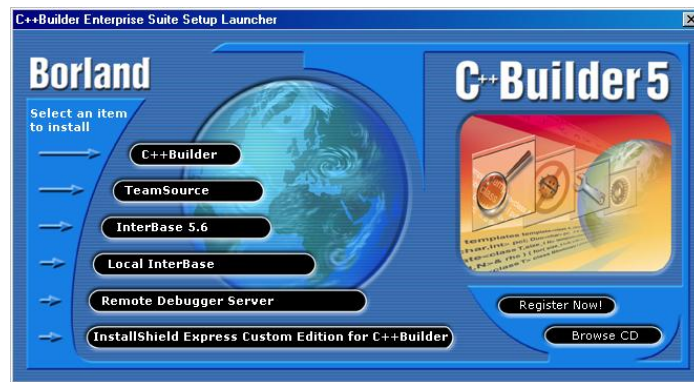
RAM 1.5 Gb

Microsoft® Windows® XP Pro SP 3

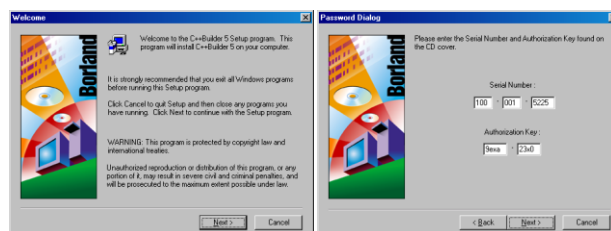
Intel® 82865G Graphics Controller

INSTALAR C++ BUILDER

En el archivo **02 Instalar Borland C++ Builder 5.txt** se comenta un poco (muy poco) del porqué compilar en Borland. Utiliza tu instalador de Borland C++ Builder 5. Realiza una instalación típica.

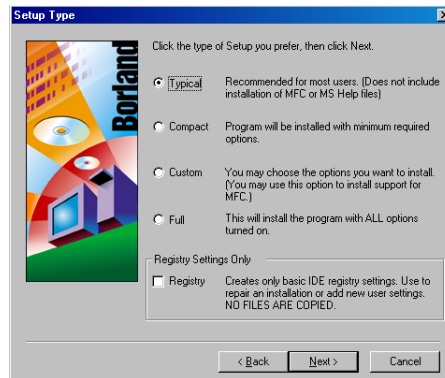
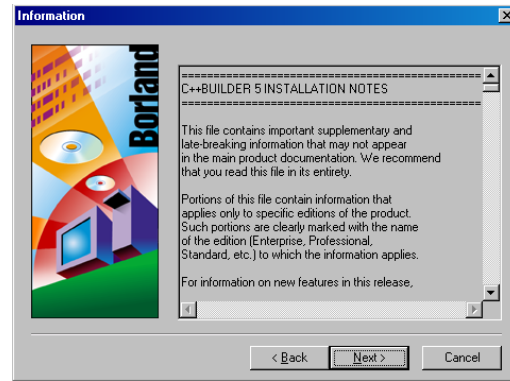
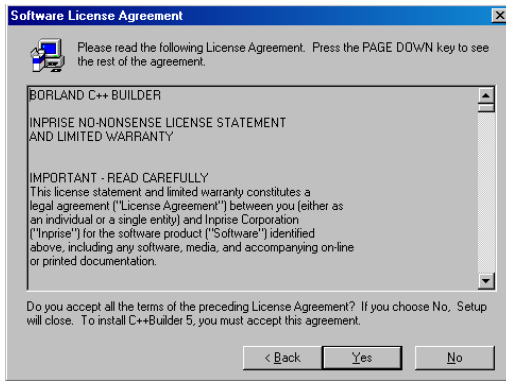


Click C++ Builder

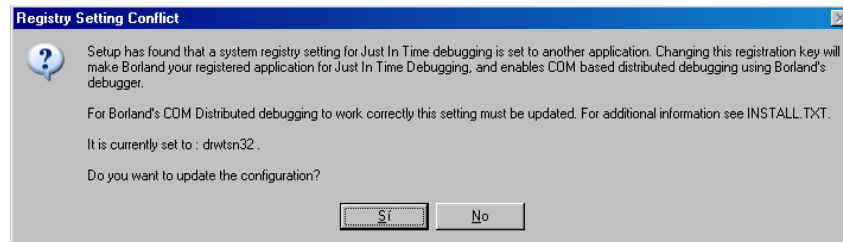


Insertar *serial number* y *Authorization Key*, con que adquiriste tu instalador de BCB

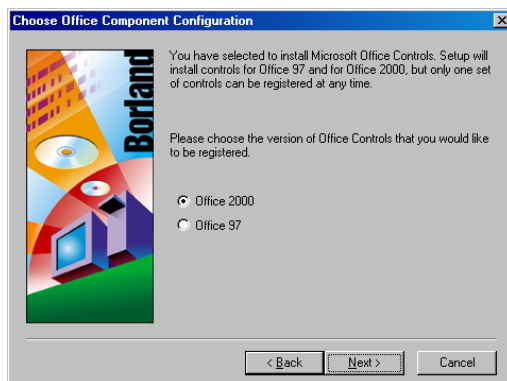
BORLAND C++ BUILDER



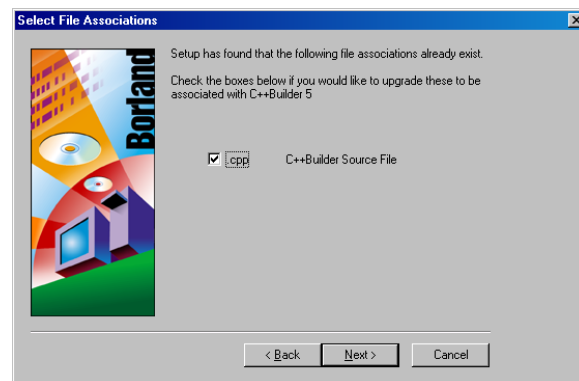
Podría aparecer:



Click **Sí**, a menos que desees lo contrario.



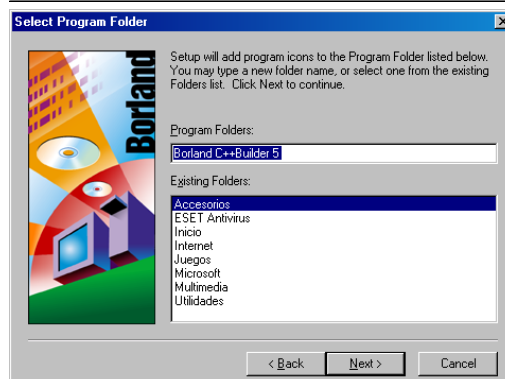
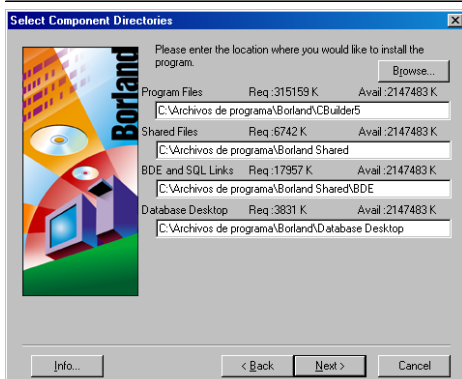
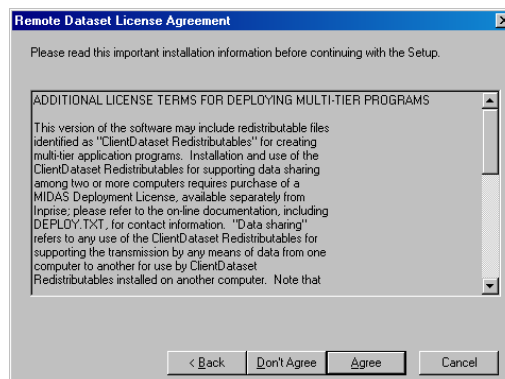
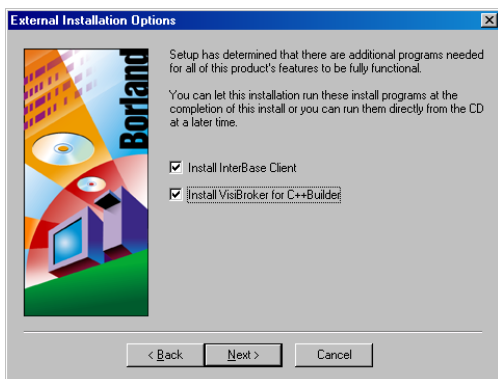
Selecciona el sistema Office correcto, Click **Next**.



Asocia las extensiones **.cpp** y click **Next**

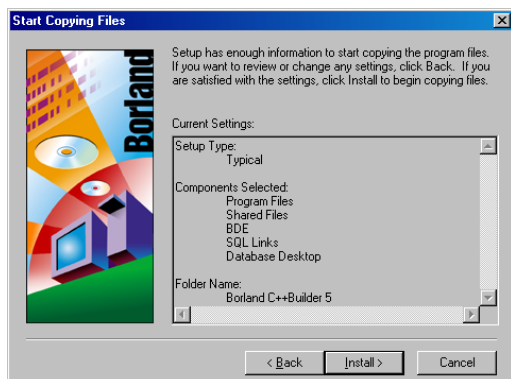
Aplicaciones Opcionales

La instalación de InterBase o Visibroker, no son estrictamente necesarias.

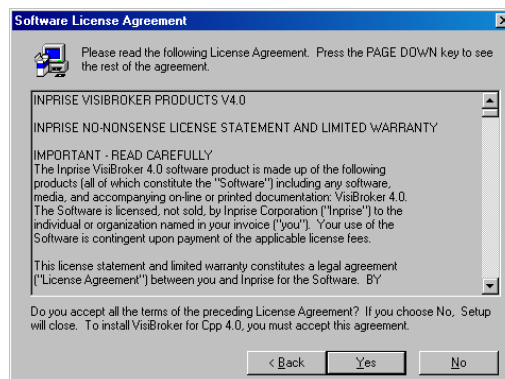
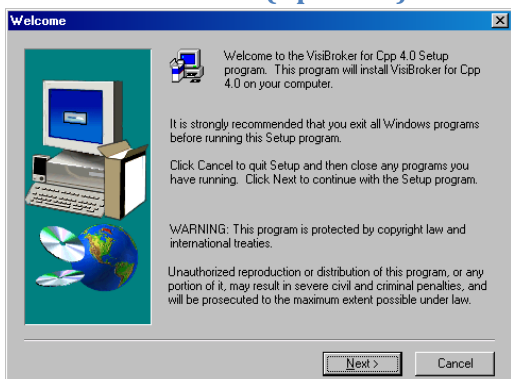


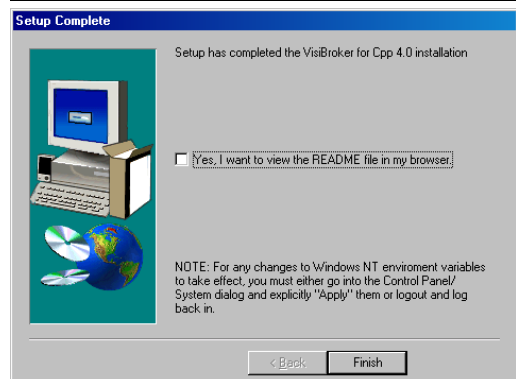
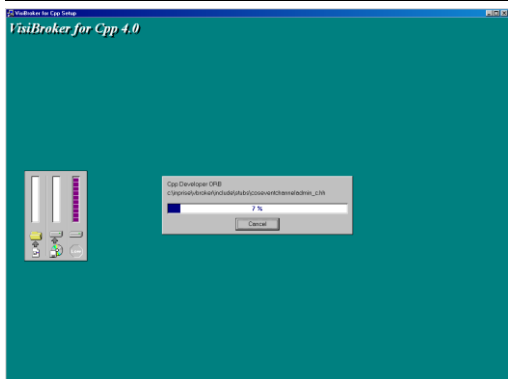
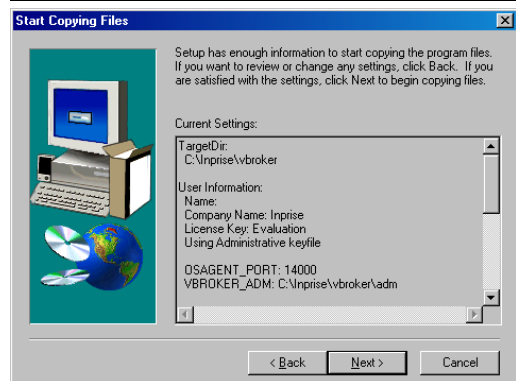
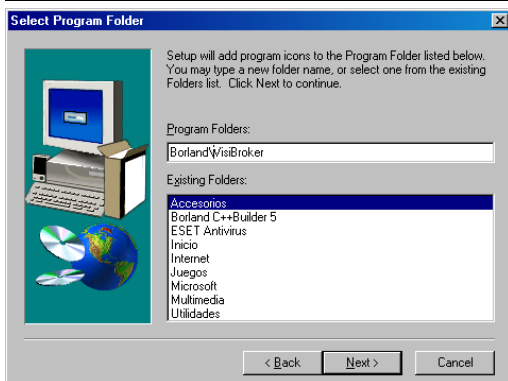
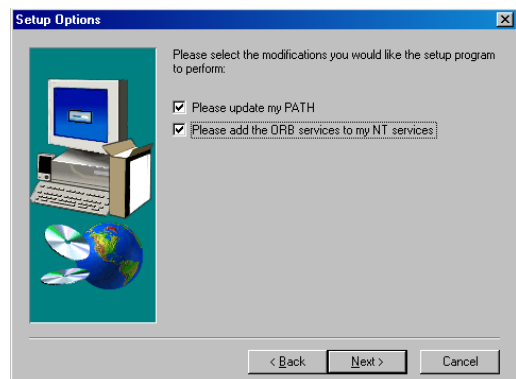
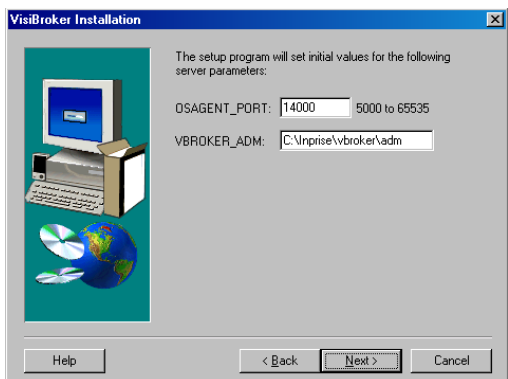
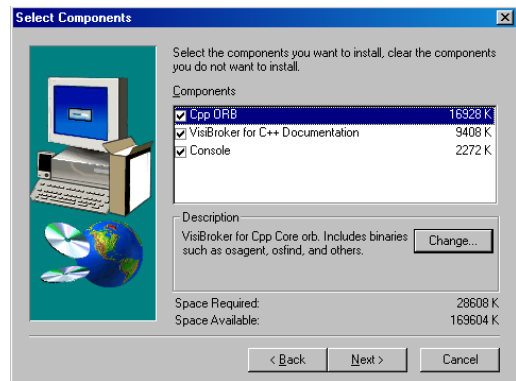
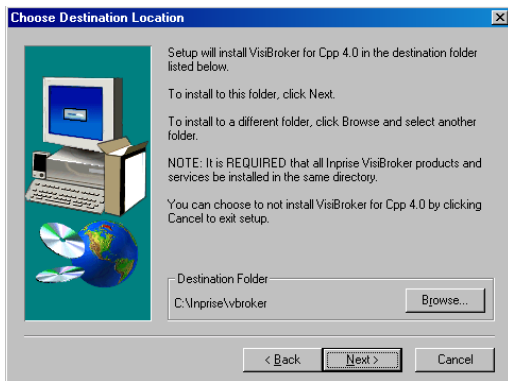
Observa los directorios donde se instalara, click Next.

Grupo del menú Inicio, Click Next

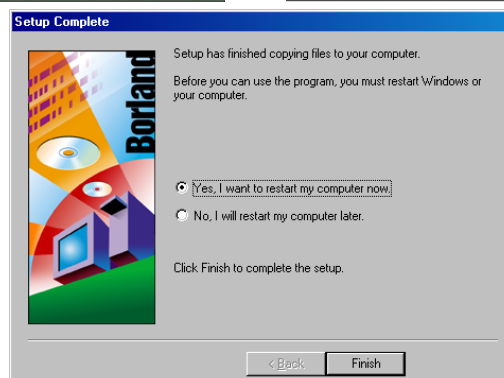
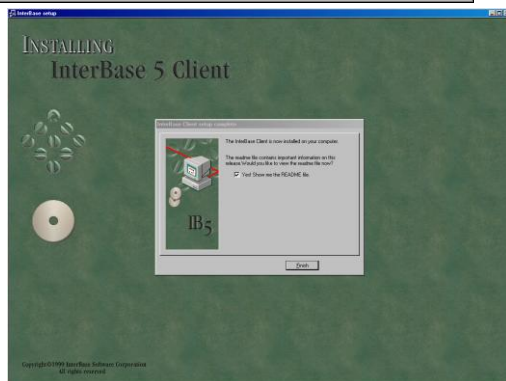
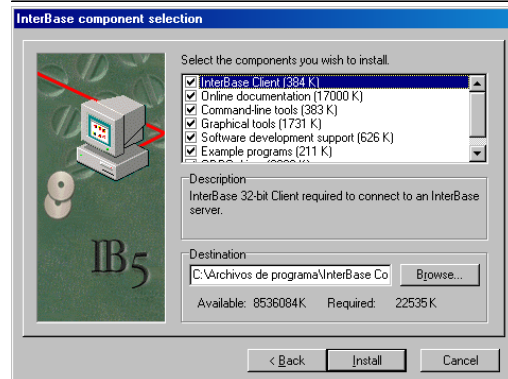
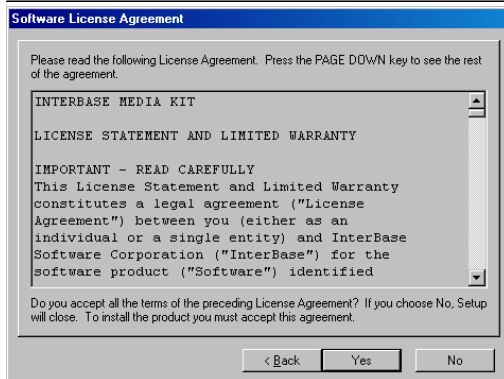
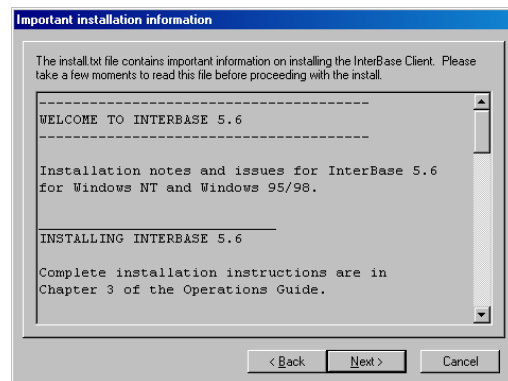
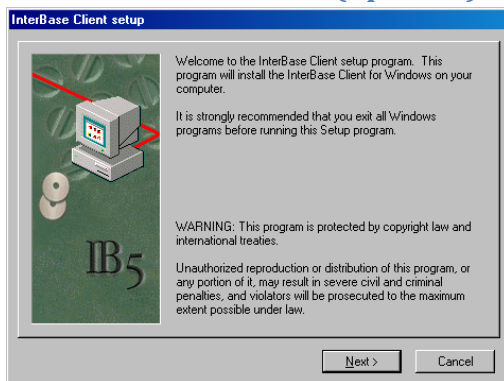


Instalar VisiBroker (Opcional)



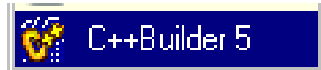


Instalar InterBase Client (Opcional)

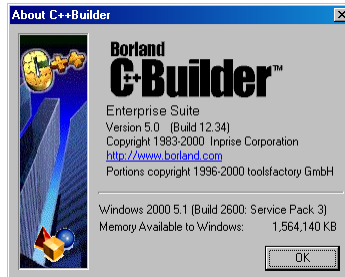


Probando la instalación de Borland® C++ Builder 5.3

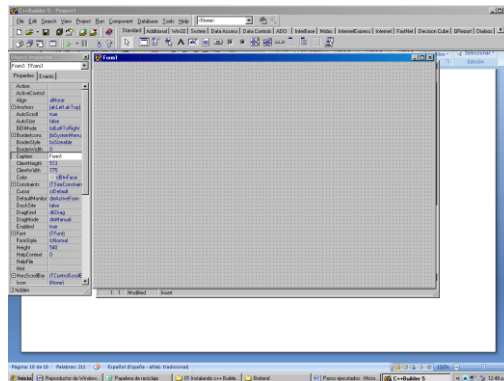
Ejecutar **C++ Builder 5**



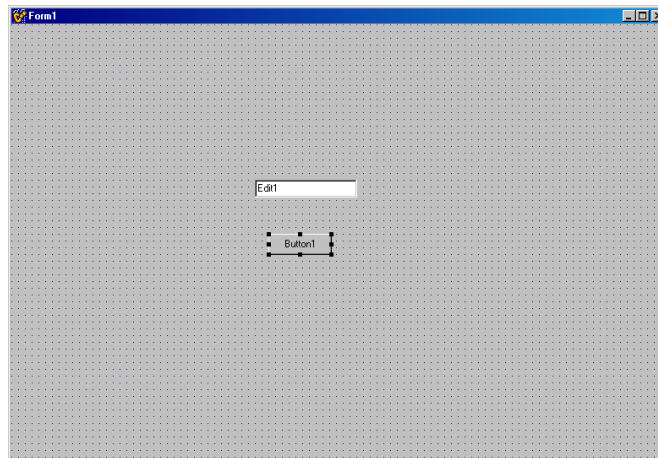
Ventana ABOUT...



Área de trabajo



Insertar un **TButton** Button1 y un **TEdit** Edit1

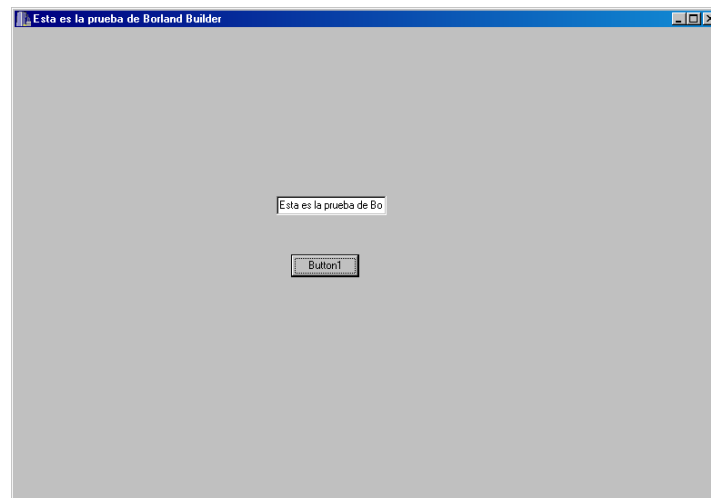


En la acción del Button1 asignar:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    Edit1->Text="Esta es la prueba de Borland Builder";
    Form1->Caption=Edit1->Text;
}
```

Ejecutar (**F9**)

Después de dar click en el Button1, debe aparecer algo como:



Guardar es opcional. Un proyecto de este **test** se encuentra en la carpeta **03 Testing c++ Builder**, o como acceso directo **03 TestingC++Builder.lnk**.

Instalar el compilador Borland C++ 5.5

Debido a situaciones con ITK que se muestran en el texto siguiente

Borland 5.5 Compiler Issues

To build Insight using the (free) Borland 5.5 compiler:

Get the Borland compiler from <http://www.borland.com> and install it

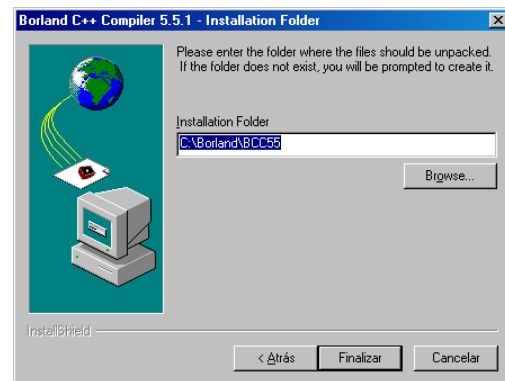
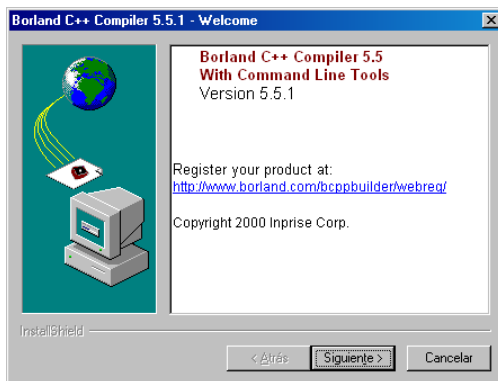
Run cmakesetup and select the Borland Makefiles in the "Build For:" drop down.

*Set the CMAKE_BUILD_TYPE to **MinSizeRel**. This is required so that the Borland linker will stay below its size limit which is around 35 meg for a tds file.*

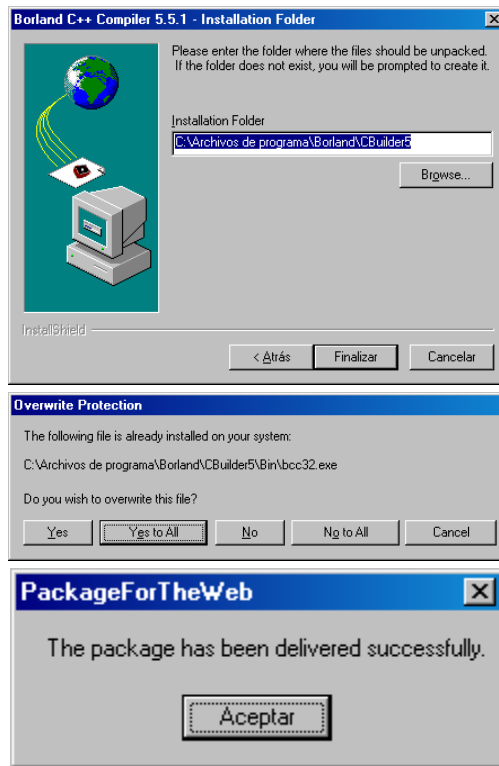
Borland 5.6 Compiler Issues

Well ITK simply does not support this compiler at all.

carpeta **04 Borland Compiler 5.5**. Al ejecutar dicho ejecutable aparece:



Debemos localizar la ruta del compilador de **Borland C++ Builder**. A saber `C:\Archivos de programa\Borland\CBUILDER5` dado que se realizó una instalación típica.



Si se desea vuelve a ejecutar el test de la carpeta **03 Testing c++ Builder**.

CMAKE, ITK Y VTK

Estos 3 elementos son independientes del compilador, pero necesarios para un proyecto conjunto.

Es necesario utilizar/generar carpetas sin espacios (de preferencia sin acentos) y lo más directo que se pueda de un directorio raíz, no utilizar guiones medios `-`, utilizar guiones bajos `_`. En teoría, si sigues la ubicación de los archivos no habrá problemas para compilar los proyectos.

CMake <http://www.cmake.org/>

CMake Versión 2.8.1, Abril 2010

ITK <http://www.itk.org/>

ITK Versión 3.16, Abril 2010

VTK <http://www.vtk.org>

VTK Versión 5.4.2, Abril 2010

Compiler Issues, Borland bcc 5.5

<http://www.itk.org/ITK/help/compilerissues.html>

No enums in templates

One of the most salient features of Borland is the lack of support for enums as numeric arguments for templates.

Exceptions in division by zero

The second salient feature is the fact that Borland throws exceptions when it encounters divisions by zero. This is actually a nice feature since it reacts to real problems in the code.

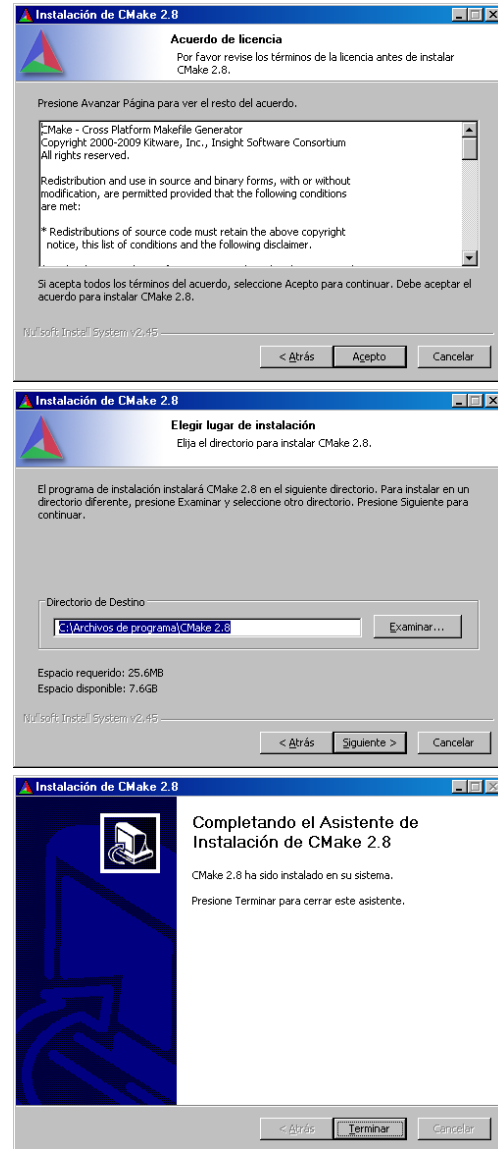
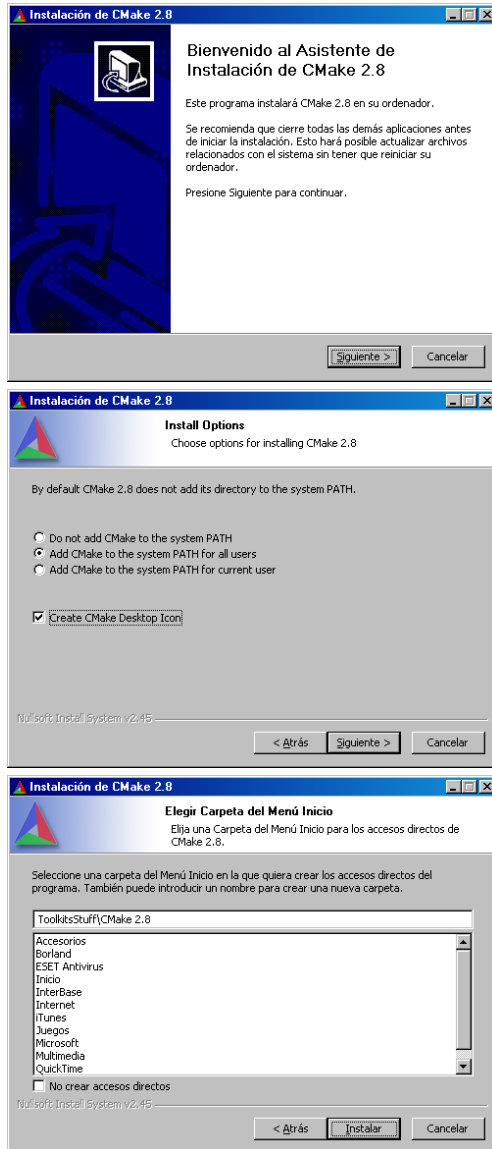
Limitation for Linking large libraries

Borland linker has a low threshold for the size of libraries it can link. The consequence of this fact is that ITK cannot be built for Debug with this compiler because the linker will not be able to manage the size of the libraries. It is recommended that you build for Release with Debug Information (RelDebInfo) when you use this compiler.

Instalar CMake

Descarga el instalador y realiza una instalación típica.

El archivo **cmake-2.8.1-win32-x86 .exe** en la carpeta **05 CMake**, es el *release* 2.8.1 de 2009. Ejecutar...



Compilar ITK

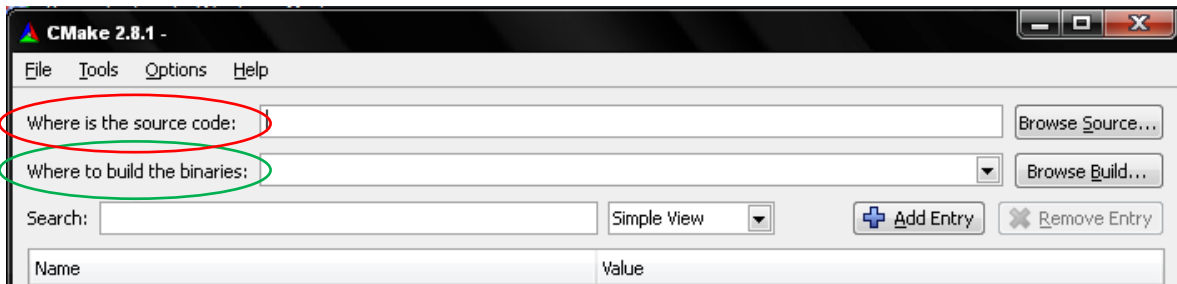
Descomprimir [ITK_CODE]

Descomprimir el archivo **InsightToolkit-3.16.0.zip** del directorio **06 ITK** en algún directorio, en este caso `[ITK_CODE]:= C:\Toolkits\Code\ITK_3_16`

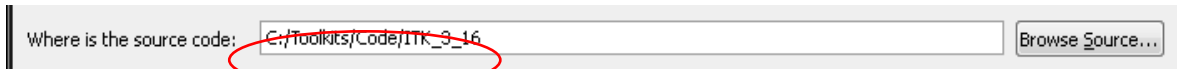
Generar Binarios ITK [ITK_Binary]



Para generar los binarios es necesario ejecutar **CMake** `CMake (cmake-gui).lnk`



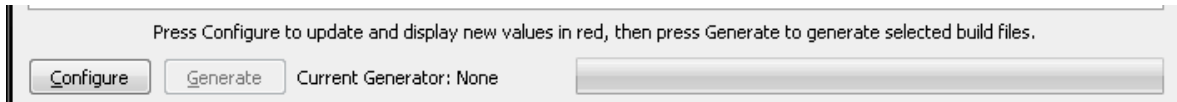
Recuerda [ITK CODE]= C:\Toolkits\Code\ITK_3_16



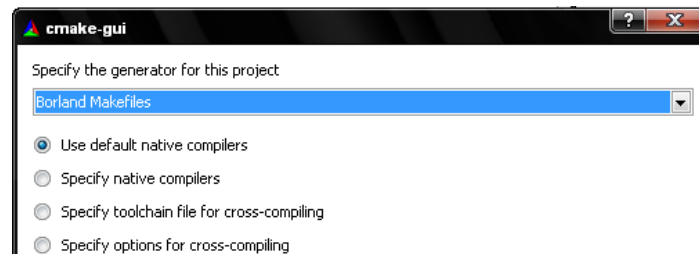
Ahora generaremos [ITK Binary]:=C:\Toolkits\Binary\ITK_3_16



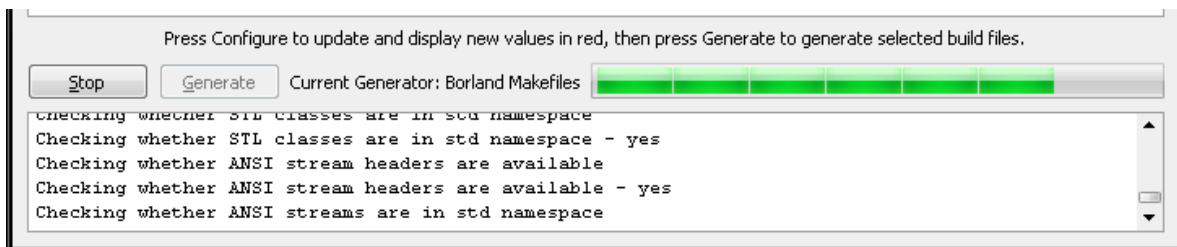
Entonces dar click en **Configure**.



Seleccionar Borland MakeFiles y use default native compilers, click Finish

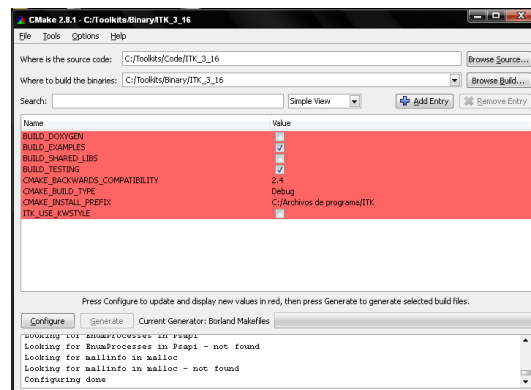


Empieza a buscar directivas...



Esperar hasta que termine.

Ahora es cuestión de configurar.



Generar Instaladores [ITK_Install]

Name	Value
BUILD_DOXYGEN	☐
BUILD_EXAMPLES	☐
BUILD_SHARED_LIBS	☐
BUILD_TESTING	☐
CMAKE_BACKWARDS_COMPATIBILITY	2.4
CMAKE_BUILD_TYPE	MinSizeRel
CMAKE_INSTALL_PREFIX	C:/Archivos de programa/ITK
ITK_USE_KWSTYLE	☐

Observa que se utiliza el
Build_Type MinSizeRel

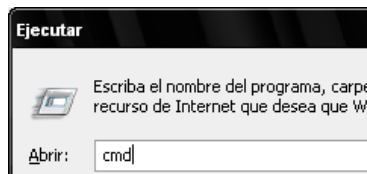
OJO:
NO Activar...

[ITK_Install]:= C:/Archivos de programa/ITK

Click Nuevamente en **configure**, Ahora Click en **GENERATE**

Compilar ITK

Menu inicio → Ejecutar



Ejecuta los siguientes comandos (o los similares dependiendo de tu ubicación [ITK_Binary])

```

C:\WINDOWS\system32\cmd.exe - make install
Microsoft Windows XP [Versión 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Oscar>cd \
C:\>cd Toolkits\Binary\ITK_3_16
C:\Toolkits\Binary\ITK_3_16>make >>..\ITK_Binary.txt
C:\Toolkits\Binary\ITK_3_16>make install >>..\ITK_Install.txt
  
```

Es el proceso que más tarda en realizarse, así que entre comando **make** y **make install** tienes tiempo suficiente y hasta de sobra para realizar otras labores.

Los archivos **ITK_Binary.txt** e **ITK_Install.txt** son archivos resumen de la compilación, debes checar que no contengan errores de compilación, puedes hacerlo en: **C:\Toolkits\Binary..**

Compilar VTK

Este proceso es similar al anterior para compilar ITK.

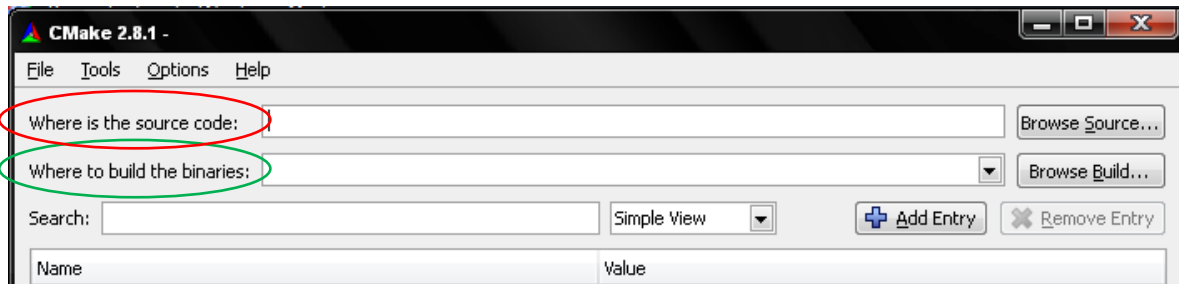
Descomprimir [VTK_CODE]

Descomprimir el archivo **vtk-5.4.2.zip** del directorio **07 VTK** en algún directorio, en este caso [VTK CODE]:= **C:\Toolkits\Code\VTK_5_4_2**

Generar Binarios VTK [VTK_Binary]



Para generar los binarios es necesario ejecutar **CMake** CMake (cmake-gui).lnk



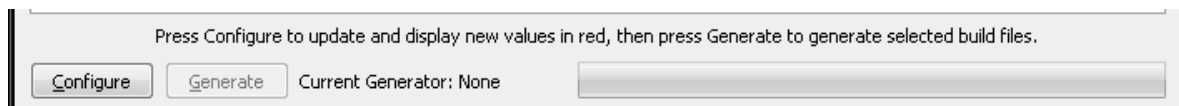
Recuerda [VTK CODE]:= C:\Toolkits\Code\VTK_5_4_2

Where is the source code: C:/Toolkits/Code/VTK_5_4_2

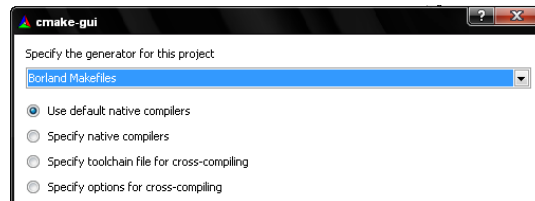
Ahora generaremos [VTK Binary]:=C:/Toolkits/Binary/VTK_5_4_2

Where to build the binaries: C:/Toolkits/Binary/VTK_5_4_2

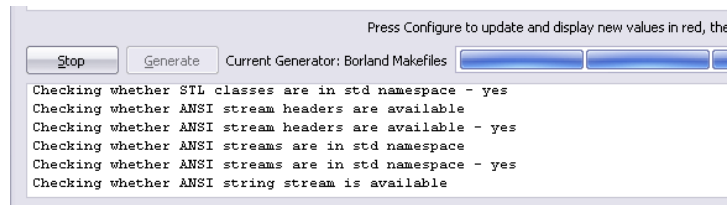
Entonces dar click en **Configure**.



Seleccionar Borland MakeFiles y use default native compilers, click Finish



Empieza a buscar directivas...



Esperar hasta que termine

Ahora es cuestión de configurar...

Generar Instaladores [VTK_Install]

Name	Value
BUILD_EXAMPLES	☐
BUILD_SHARED_LIBS	☒
BUILD_TESTING	☐
CMAKE_BACKWARDS_COMPATIBILITY	2.4
CMAKE_BUILD_TYPE	MinSizeRel
CMAKE_INSTALL_PREFIX	C:/Archivos de programa/VTK
VTK_DATA_ROOT	VTK_DATA_ROOT-NOTFOUND
VTK_USE_GEOVIS	☒
VTK_USE_INFOVIS	☒
VTK_USE_N_WAY_ARRAYS	☐
VTK_USE_PARALLEL	☐
VTK_USE_RENDERING	☒
VTK_USE_VIEWS	☒
VTK_WRAP_JAVA	☐
VTK_WRAP_PYTHON	☐
VTK_WRAP_TCL	☐

OJO:
Activar...

Observa que se utiliza el *Build_Type*
MinSizeRel

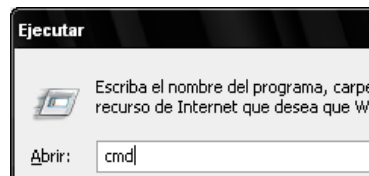
[VTK_Install]:= C:/Archivos de programa/VTK

Click Nuevamente en **configure**

Ahora Click en **GENERATE**

Compilar VTK

Menu inicio → Ejecutar



Ejecuta los siguientes comandos (o los similares dependiendo de tu ubicación [VTK_Binary])

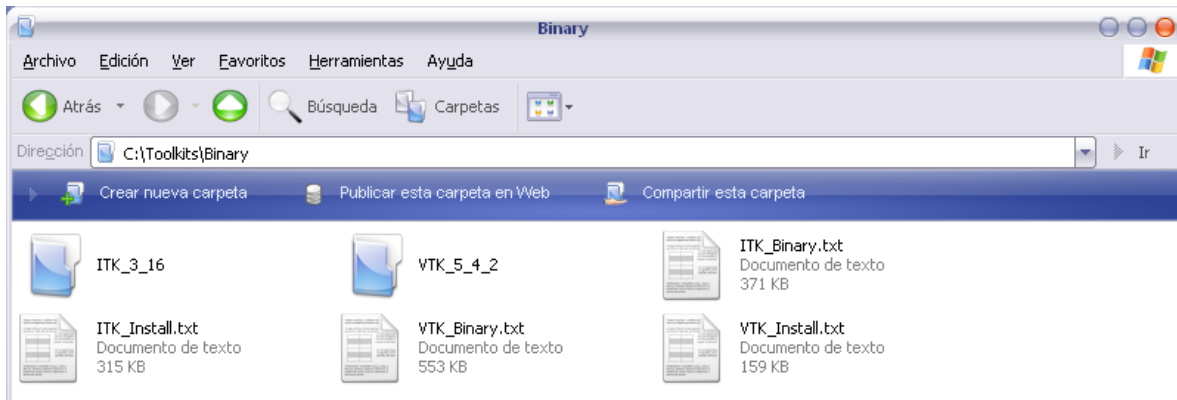
```

C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Versión 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Oscar>cd \
C:\>cd Toolkits\Binary\UTK_5_4_2
C:\Toolkits\Binary\UTK_5_4_2>make >>..\UTK_Binary.txt
C:\Toolkits\Binary\UTK_5_4_2>make install >>..\UTK_Install.txt
C:\Toolkits\Binary\UTK_5_4_2>
  
```

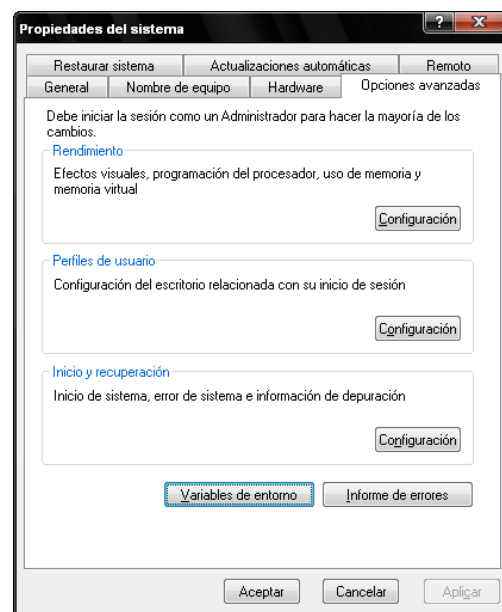
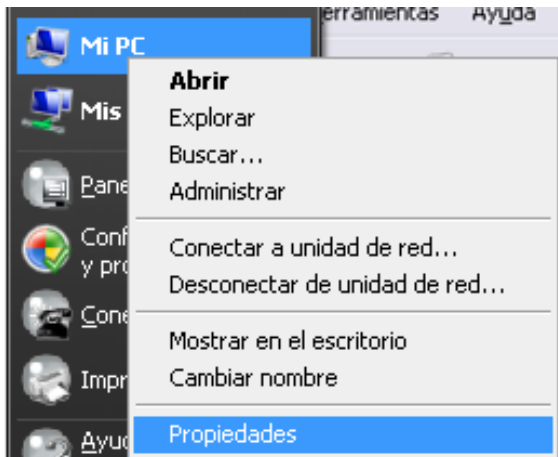
Es el proceso que más tarda en realizarse, así que entre comando **make** y **make install** tienes tiempo de continuar con otras labores.

Los archivos **ITK_Binary.txt**, **ITK_Install.txt**, **VTK_Binary.txt** y **VTK_Install.txt** son archivos resumen de la compilación, debes checar que no contengan errores de compilación, puedes hacerlo en: **C:\Toolkits\Binary**.

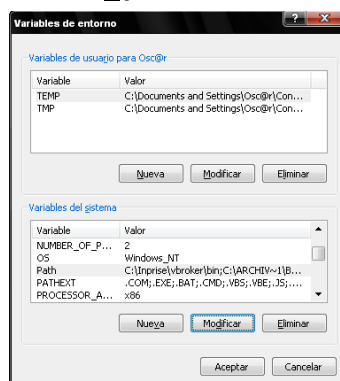


Para finalizar, deberás añadir la carpeta [VTK_Install]\bin al PATH de las variables de entorno de WINDOWS, debido a que vamos a utilizar las librerías compartidas (SHARED LIBRARIES). Esta acción es necesaria para poder compilar sin tanto problema los proyectos conjuntos VTK/Builder/ITK.

Abrir Propiedades de MI PC y en la pestaña Opciones Avanzadas Click en el botón de Variables de entorno

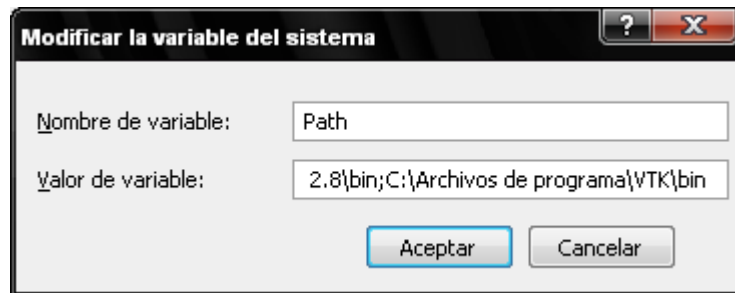


Buscar la variable **Path**, click en **Modificar**.



AÑADIR, cuidado de **NO BORRAR** nada.

Añadir el directorio [VTK_Install]\bin=C:\Archivos de programa\bin, o donde quiera que se hayan instalado las librerías VTK*.dll. Click en **Aceptar**



BORLAND C++ BUILDER® & VTK

Para generar proyectos con VTK es necesario instalar un paquete (*Borland Package*) para visualización.

Instalar Paquete VTKBorlandRenderWindow en Borland

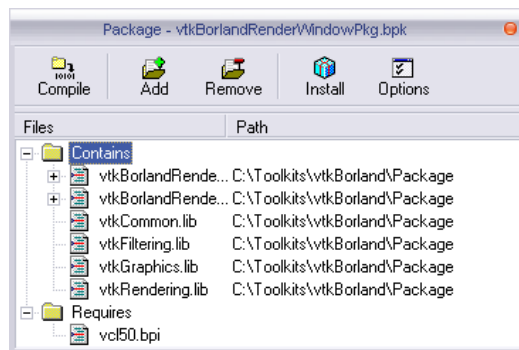
Descomprimir el archivo **08 vtkBorland.zip** en alguna carpeta, digamos **C:\Toolkits**

Entrar a la carpeta **C:\Toolkits\vtkBorland\Package** y ejecutar **vtkBorlandRenderWindowPkg.bpk**

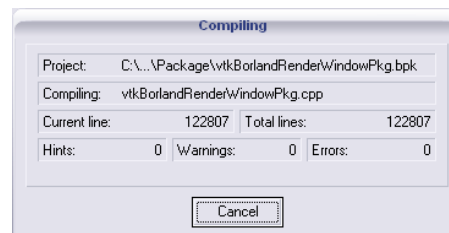


vtkBorlandRenderWindowPkg.bpk

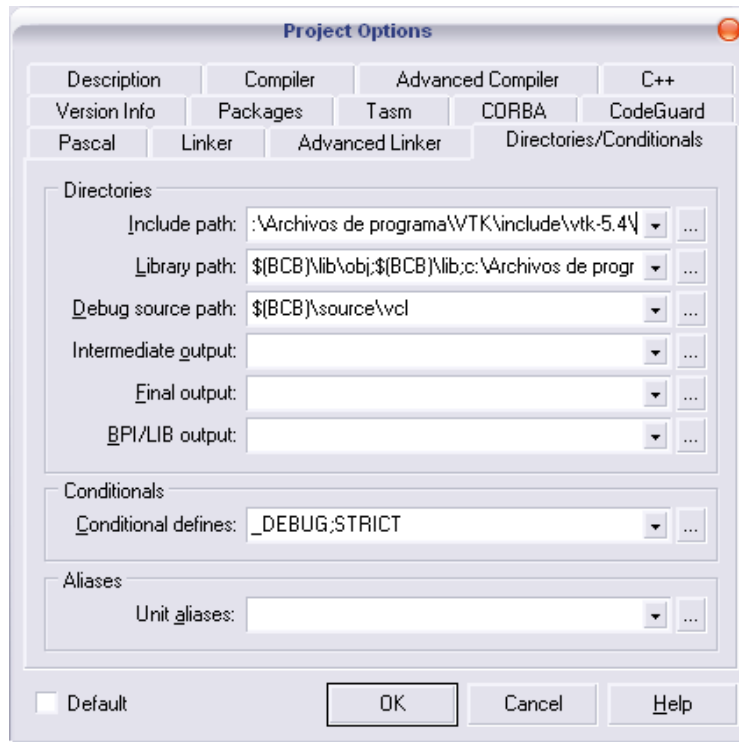
Dicha acción abrirá Builder...



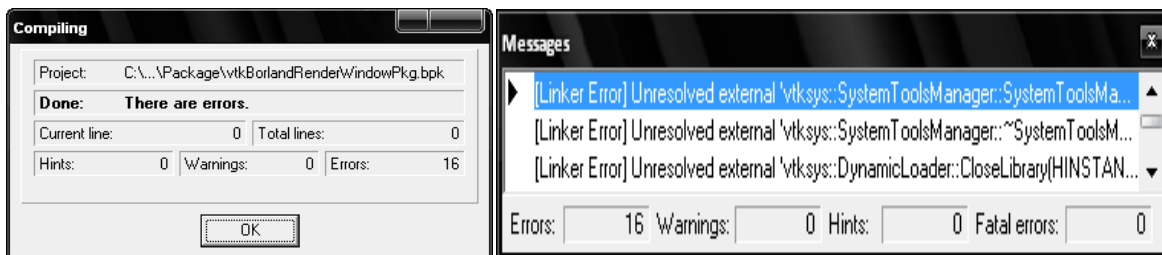
Click en **Compile**, es probable que marque algunos errores, para ello muestro posibles soluciones a continuación.



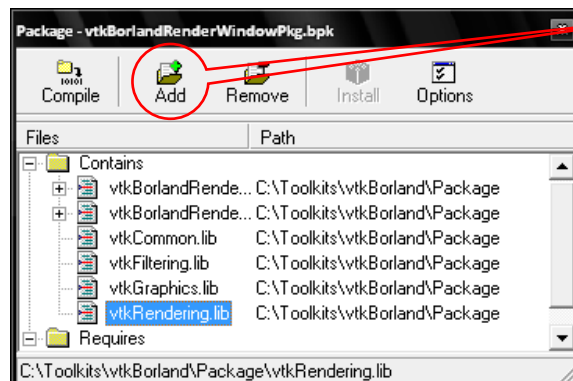
A decir verdad, en mis intentos me marcó errores... Y se colocó el cursor en la línea `#include"vtkcommand.h"` lo que me hace suponer que me falta añadir algunos directorios de directivas. Para resolver di click en **Options.**, y en **Directories/Conditionals**, verifique que en **Include Path** estuviera el directorio `[VTK_Install]\include\vtk-5.4\`, y en **Library Path** estuviera `[VTK_Install]\lib\vtk-5.4\`. el **5.4** es debido a la versión, si cambias la versión de VTK, verifica los directorios.



Marcó errores de LINKER. Por lo que se requiere otras acciones

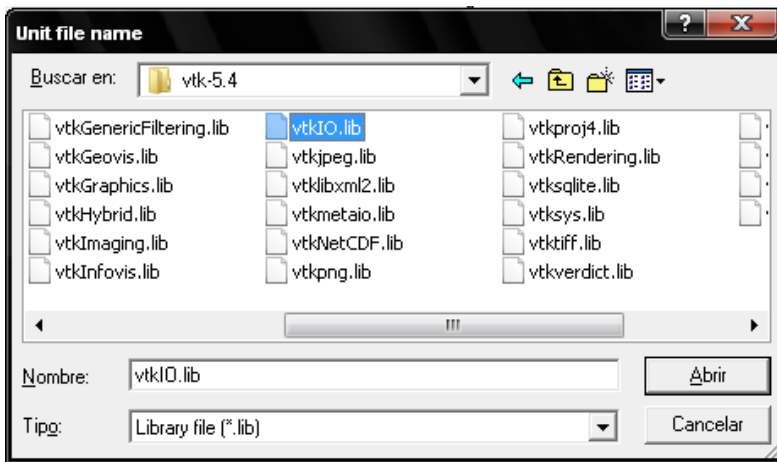


Estos errores de LINKER se resuelven añadiendo librerías directamente. Esto es en **Add**

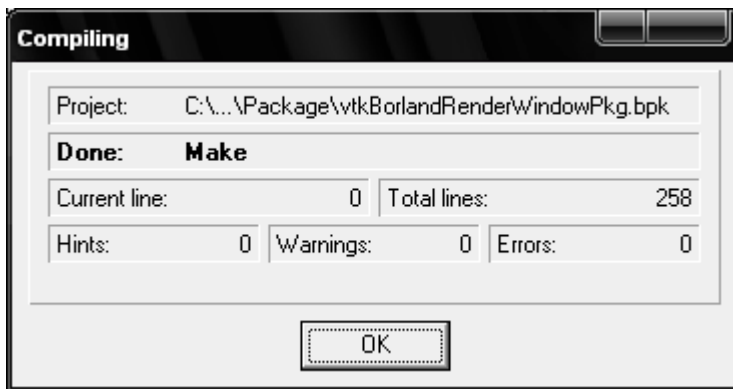


Buscar el directorio y archivo tipo .lib,
[VTK_Install]\lib\vtk-5.4\vtksys.lib

Y también el archivo [VTK_Install]\lib\vtk-5.4\vtkIO.lib



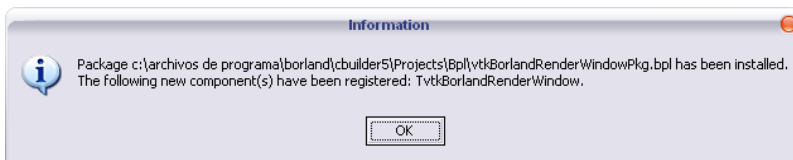
Compilar nuevamente. Si observas la siguiente ventana, ya está listo para instalar.



Una vez que tienes compilado el paquete solo falta

instalarlo, para ello dar click en **Install**

El siguiente mensaje debe aparecer.



Enseguida cierra y guarda. **File->Close All**, elige **Save**

Debe aparecer el **icono** del componente en la paleta **Samples**



Si acaso cambiaste de versión de VTK y continua marcando errores de LINKER, puedes buscar dependencias en el archivo *C:\Archivos de programa\VTKLib\vtk-5.4\VTCLibraryDepends.cmake* y a partir de las librerías involucradas en los errores y la información contenida en el archivo añadir las librerías correspondientes.

EJEMPLO:

Anteriormente uno de los errores que marcaba era el siguiente...

[Linker Error] Unresolved external

'vtkXMLMaterial::GetNumberOfVertexShaders()' referenced from C:\ARCHIVOS DE PROGRAMA\VTK\LIB\VTK-5.4\VTKRENDERING.LIB|vtkOpenGLProperty,.

La librería involucrada se encuentra resaltada. Ahora abro el archivo de dependencias, busco las dependencias de la librería vtkRendering y observo la siguiente línea:

```
SET("vtkRendering_LIB_DEPENDS"
    general;
    vtkGraphics;
    general;
    vtkImaging;
    general;
    vtkIO;
    general;
    vtkfgl;
    general;
    vtkfreetype;
    general;
    import32;")
```

Se encuentran resaltadas las dependencias, aunque probé con todas ellas, fue **vtkIO.lib** quien eliminó el error antedicho.

Demo vtkBorland



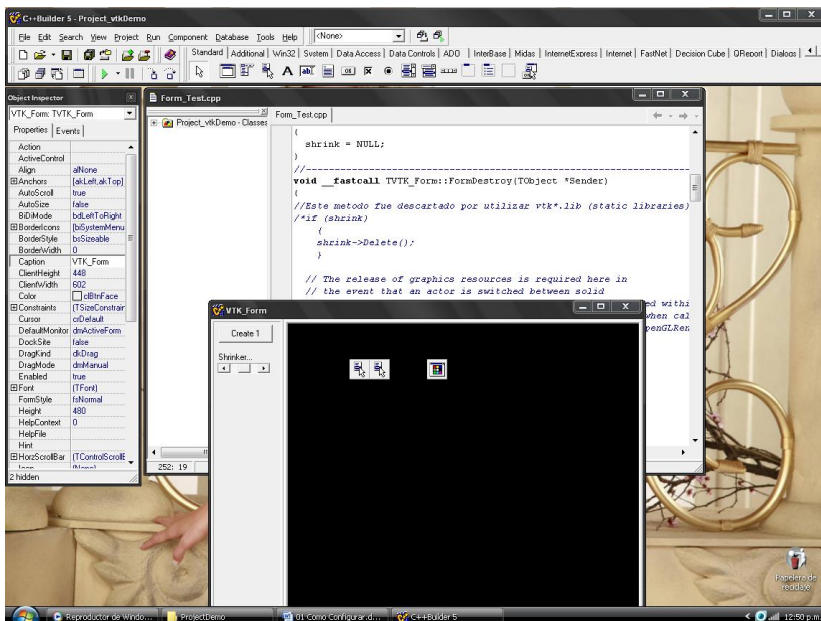
08.1 vtkDemo.exe

Es un proyecto sencillo, como ejemplo se tiene el ejecutable . Utiliza el componente de *vtkBorlandRenderWindow*. Los archivos del proyecto se han creado en C++ Builder 6 por *Dean Inglis* y probados en BCB 5 por el mismo, han sido ligeramente modificados por *Oscar*



Project_vtkDemo.bpr

Gtz Luna para este manual. El proyecto se encuentra en **C:\Toolkits\vtkBorland\ProjectDemo**. (Suponiendo que estas siguiendo los directorios de extracción antedichos)

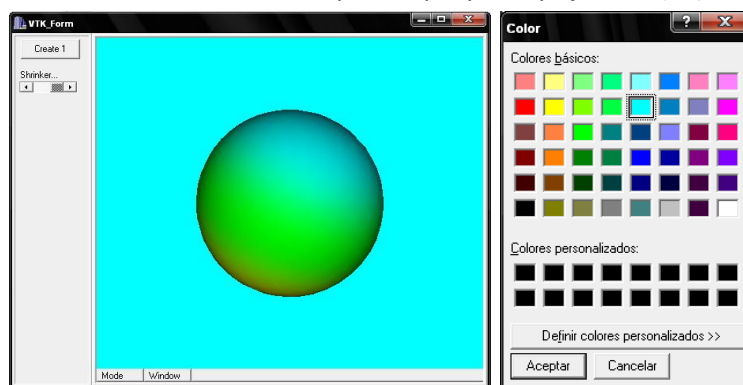


Los comentarios en español fueron escritos por su servidor: Oscar Gutiérrez Luna.

Los comentarios en Inglés son los propios de los proyectos de Dean Inglis.

Fueron cambios mínimos los realizados por su servidor.

El proyecto de demostración es muy simple y ayuda para empezar a trabajar con el componente *vtkBorlandRenderWindow*. Compilar el proyecto y ejecutar (**F9**)...



Debes verificar que los directorios **include** y **lib** de **vtk** se encuentren asignados correctamente.

Errores de LINKER pueden ser resueltos como lo explicado en la nota de la página anterior.

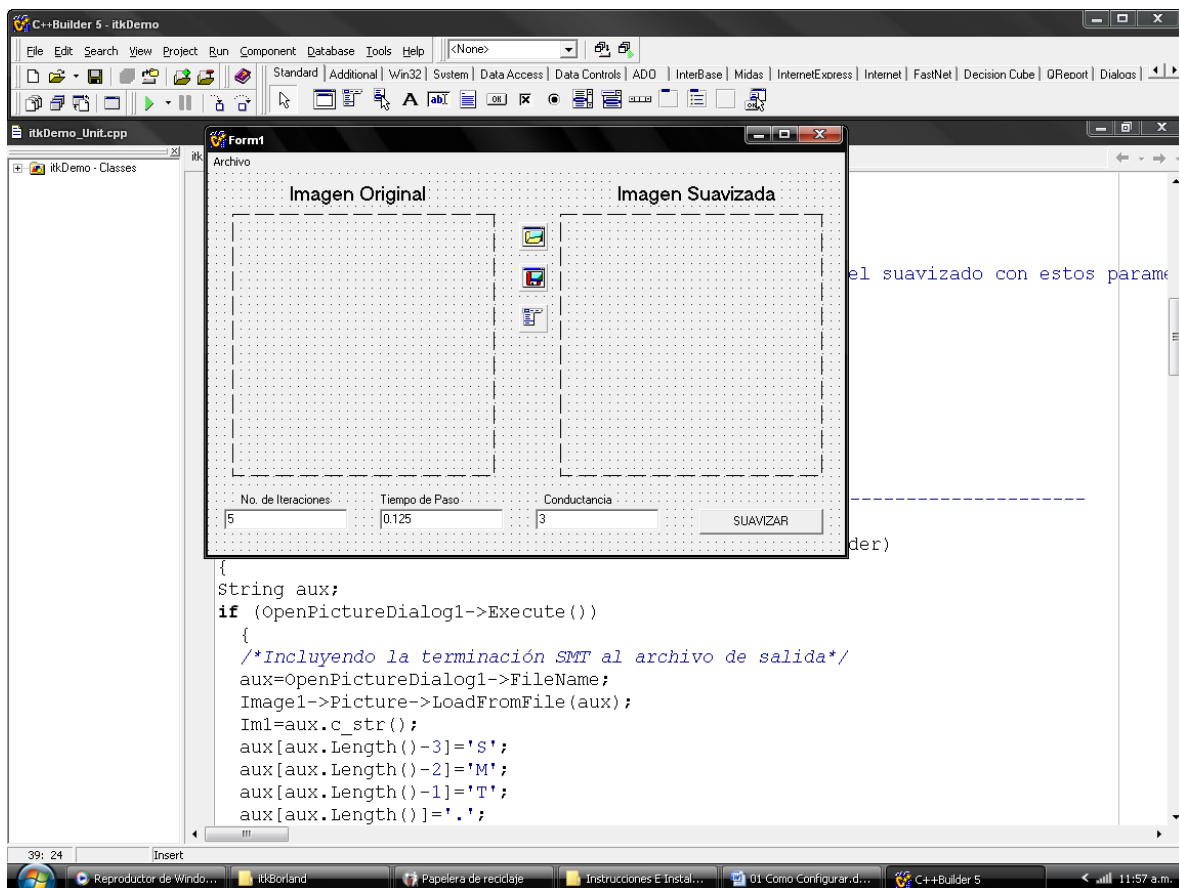
Para comprender las características de este Demo en cuanto a movimiento y visualización, es momento de leer **bibliografía** de **VTK**.

BORLAND C++ BUILDER® & ITK

El primer proyecto que compilaremos con Borland C++ Builder® & ITK es uno de los demos de suavizado que preserva bordes que contiene el archivo **06.1 itkSoftwareGuide-2.4.0.pdf** en la página 197, **II User's Guide → Filtering → Smoothing Filters → Edge Preserving Filters → CurvatureAnisotropicDiffusionImageFilter**. Aunque el demo de la guía de software de itk es para realizarse en modo consola, ha sido trasladado a un proyecto Borland Builder con Interfaz Gráfica de Usuario (**GUI** por sus siglas en Inglés, **Graphic User Interface**), Analice las similitudes y diferencias. La diferencia principal es que el **PIPELINE** ejecutado en la función **main** (modo consola), es ahora la función **suavizar** (Interfaz gráfica de usuario), ejecutada a través del botón **SUAVIZAR**.

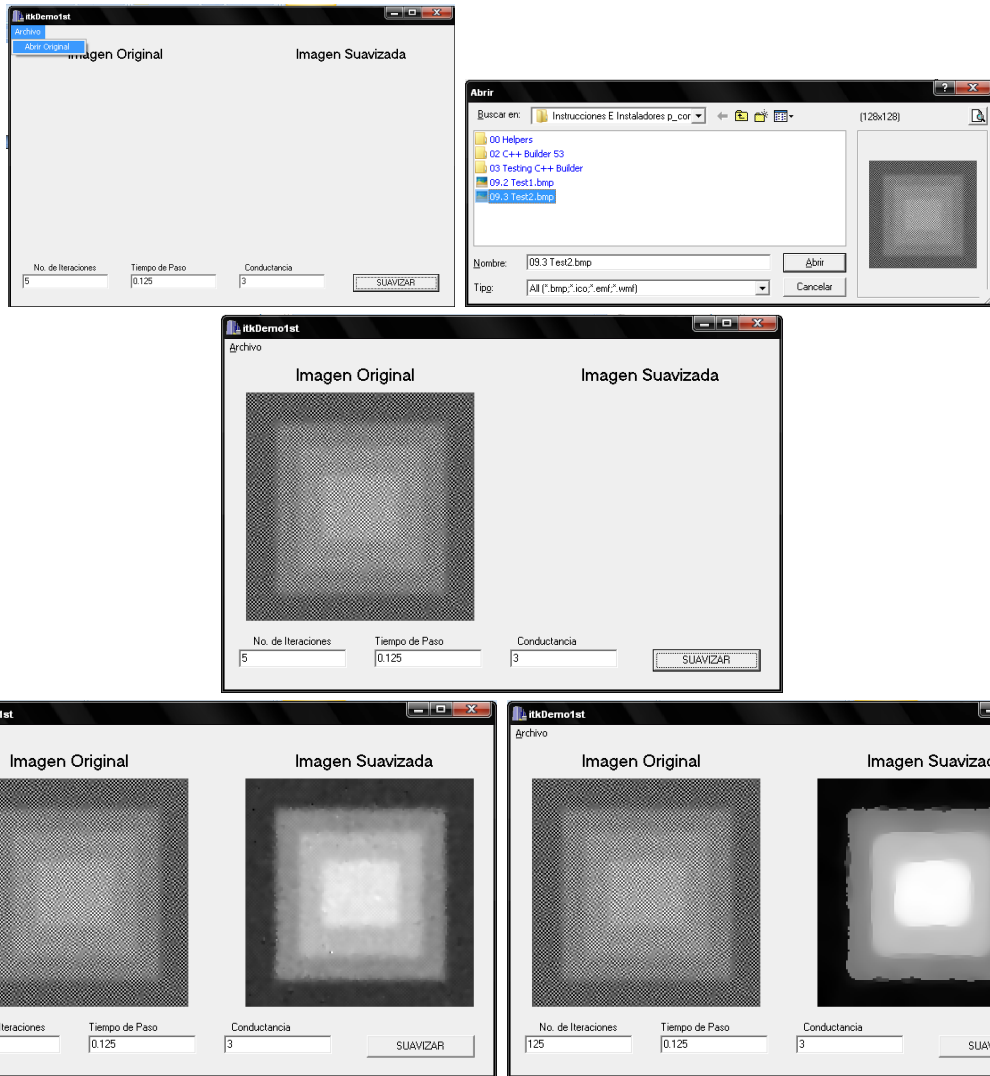
Descomprimir el archivo **09 itkBorland.zip** en alguna carpeta, digamos **C:\Toolkits**

Ejecutar  de **C:\Toolkits\itkBorland** aparecerá:

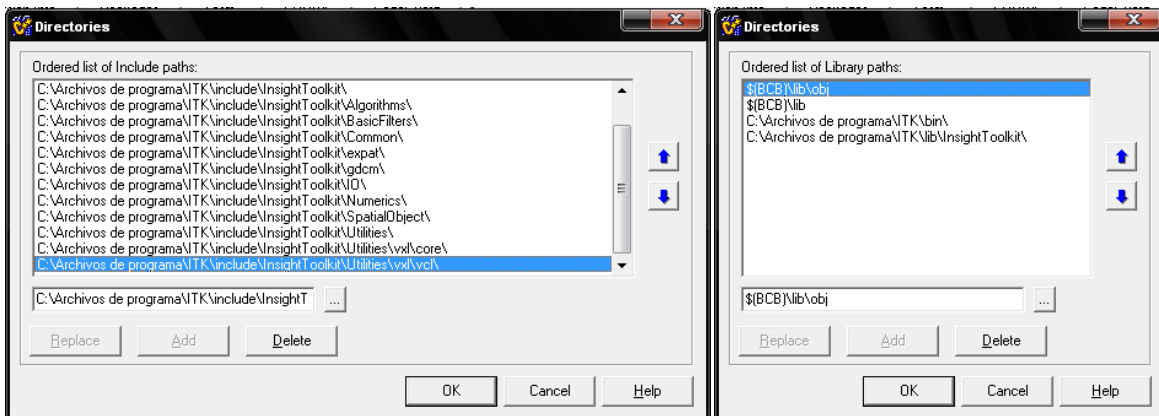


Compilar el proyecto y ejecutar (**F9**)...

Cargar una imagen, **Archivo->Abrir Original...**



En este caso no me marcó ningún error a pesar de la actualización de ITK, debido a que ITK no añade a las carpetas ningún indicador de la versión, en caso de tener problemas, verificar que los directorios **include** y **lib** estén incluidos en las carpetas de búsqueda del proyecto.



Para comprender las características de este demo, es momento de leer **bibliografía** de ITK. Un elemento muy completo es **06.1 itkSoftwareGuide-2.4.0.pdf**.

BORLAND C++ BUILDER®, ITK & VTK

En el demo anterior se utilizaron visualizadores de *Builder* el cual soporta imágenes 2-dimensionales, la finalidad de este manual es utilizar la visualización 3D que ofrece VTK, y el procesamiento que ofrece ITK. Dado que este manual surge como parte del proyecto “**MARVIN_LS**: plugin de segmentación por *level sets* para el sistema de visualización y procesamiento 3D: **Marvin**”, la siguiente aplicación utiliza muchos de los elementos utilizados en generar **MARVIN_LS**, aunque la finalidad de este manual, es mostrar como configurar los 3 elementos para un proyecto conjunto, para después ser capaces de generar aplicaciones desde básicas hasta complejas, aquí se ofrece este proyecto un tanto complejo, el cuál calcula y muestra las derivadas direccionales de una imagen 3D, así como las magnitudes de las diferencias direccionales y la magnitud del gradiente.

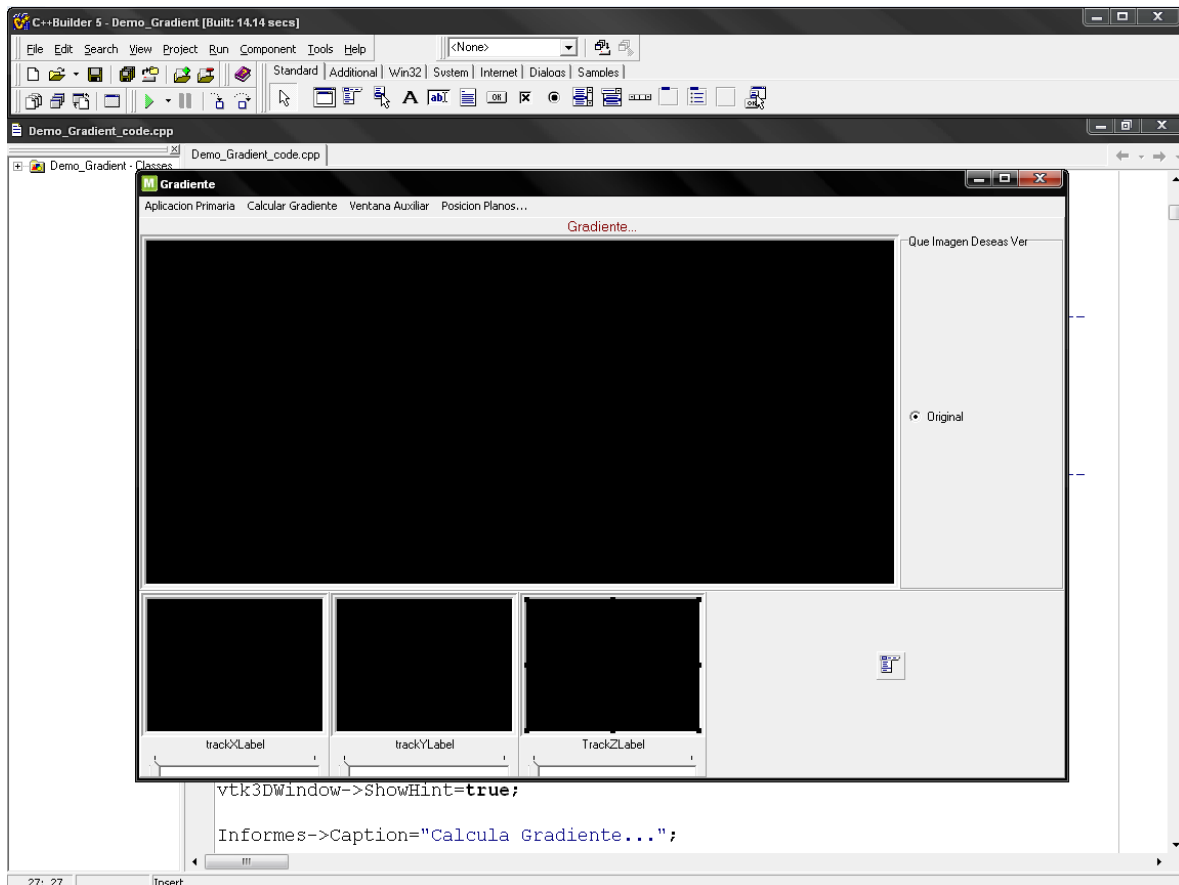
Descomprimir el archivo **10 Gradiente_VtkItkBorland.zip** en alguna carpeta, digamos **C:\Toolkits**

Entrar a la carpeta **C:\Toolkits\Vtk_Itk_Borland\Demo_Gradient_Basico** ejecutar **Demo_Gradient.bpr**.



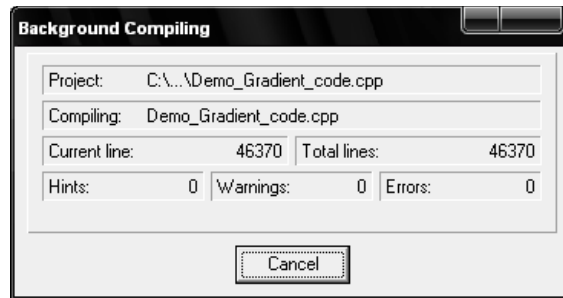
Demo_Gradient.bpr

Dicha acción abrirá Builder...

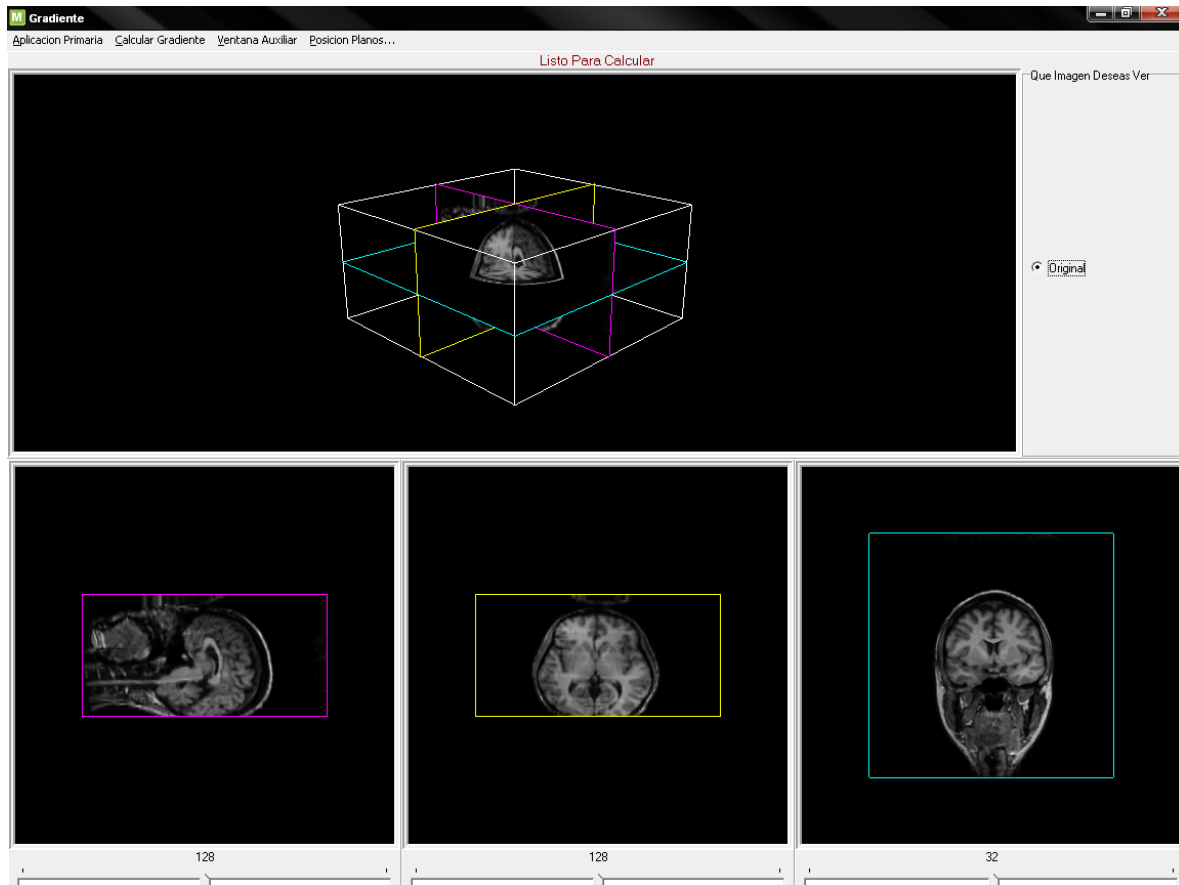


Compilar el proyecto y ejecutar (**F9**)...

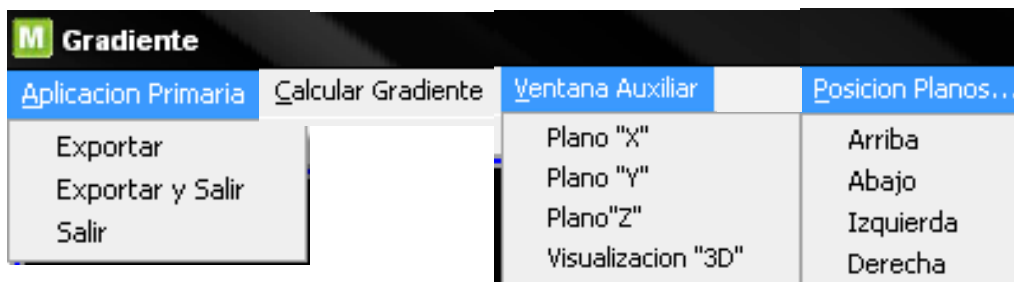
BORLAND C++ BUILDER



Una vez que pudo compilar y ejecutar...



El uso de esta aplicación no es complicada (espero!!!), tiene como casi toda GUI su barra de menus.



Aplicación Primaria

Debe su existencia a la posibilidad de ser un plugin.

Exportar, Exporta los datos resultados de regreso a la Aplicación Primaria

Exportar y Salir, Exporta los datos resultados de regreso a la Aplicación Primaria y sale de la aplicación actual.

Salir, Termina su ejecución sin enviar datos a la Aplicación Principal.

Calcular Gradiente

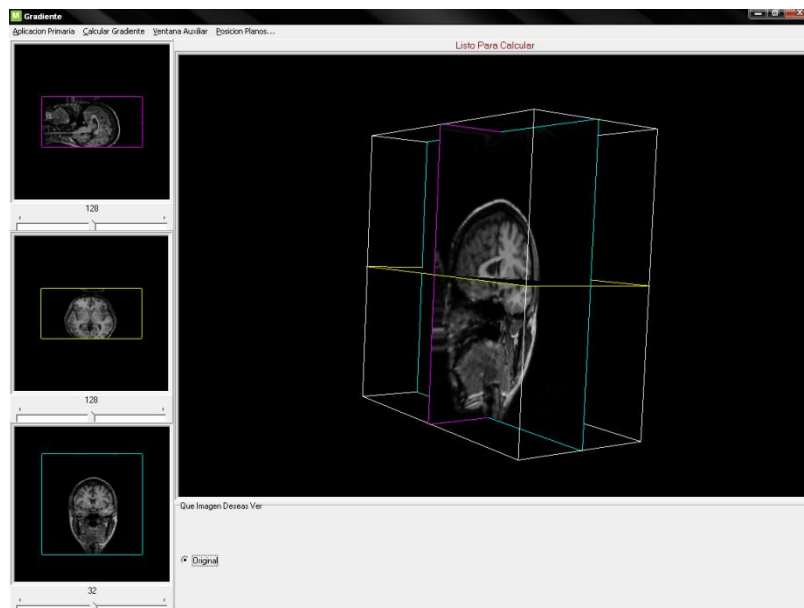
Es la actividad primaria de esta aplicación, y al ejecutarse genera 7 resultados distintos.

- Derivadas Parciales
 1. Derivada parcial por diferencias hacia adelante en la dirección X.
 D_x
 2. Derivada parcial por diferencias hacia adelante en la dirección Y.
 D_y
 3. Derivada parcial por diferencias hacia adelante en la dirección Z.
 D_z
- Magnitud de las derivadas parciales
 4. Magnitud de **D_x** , **$|D_x|$**
 5. Magnitud de **D_y** , **$|D_y|$**
 6. Magnitud de **D_z** , **$|D_z|$**
- Magnitud del Gradiente
 7. **$|D_x| + |D_y| + |D_z|$** , es decir, es calculado con la norma 1.



Posición Planos

Permite posicionar los planos 2D en las regiones indicadas, por default se colocan Abajo.



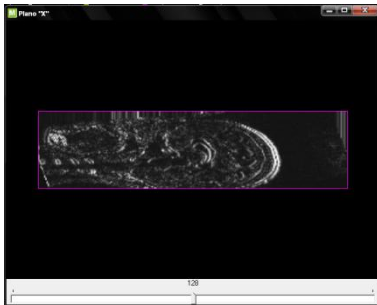
Posición Planos → Izquierda

Ventana Auxiliar

Abre en una ventana flotante la imagen activa.

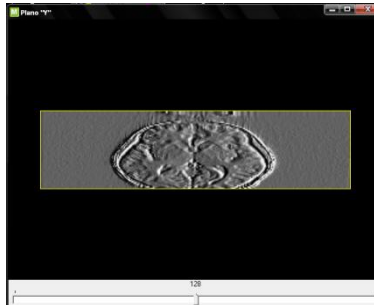
Plano "X"

$|Dy|$



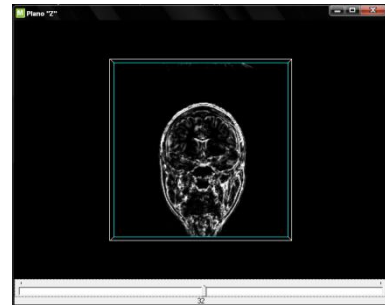
Plano "Y"

Dx

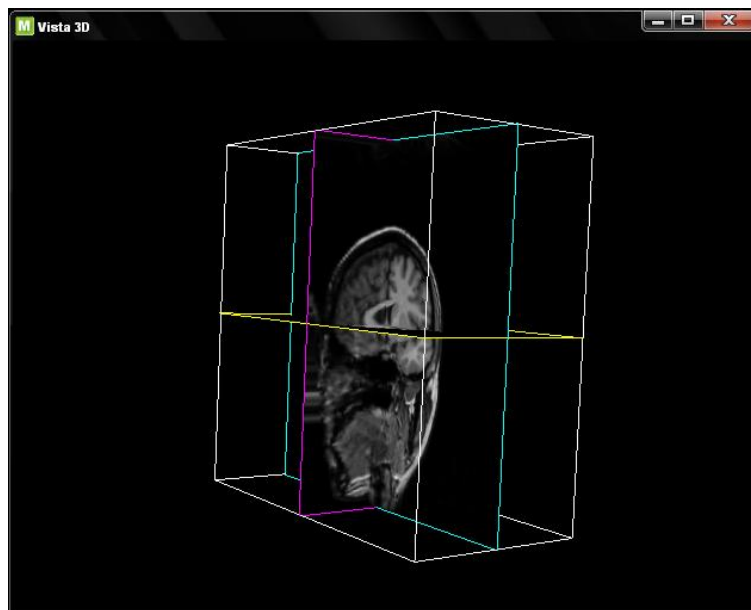


Plano "Z"

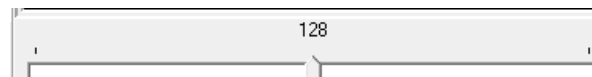
Magnitud del Gradiente



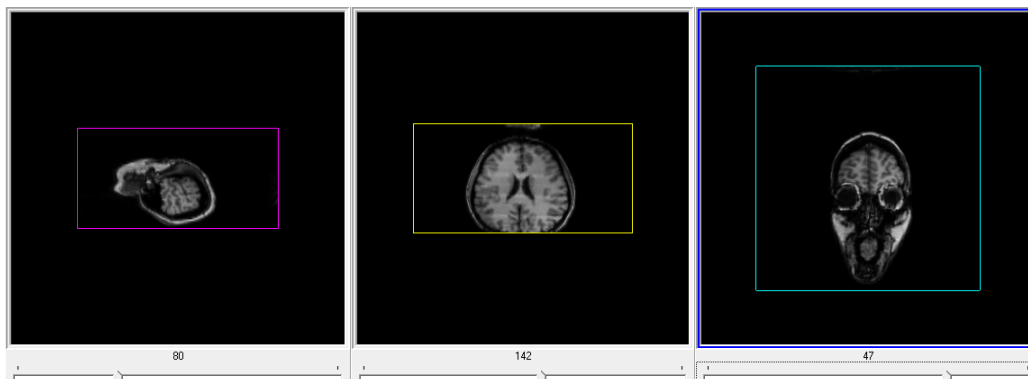
Visualización 3D



Selector de Rebanadas...



Permite seleccionar la rebanada en el plano correspondiente.

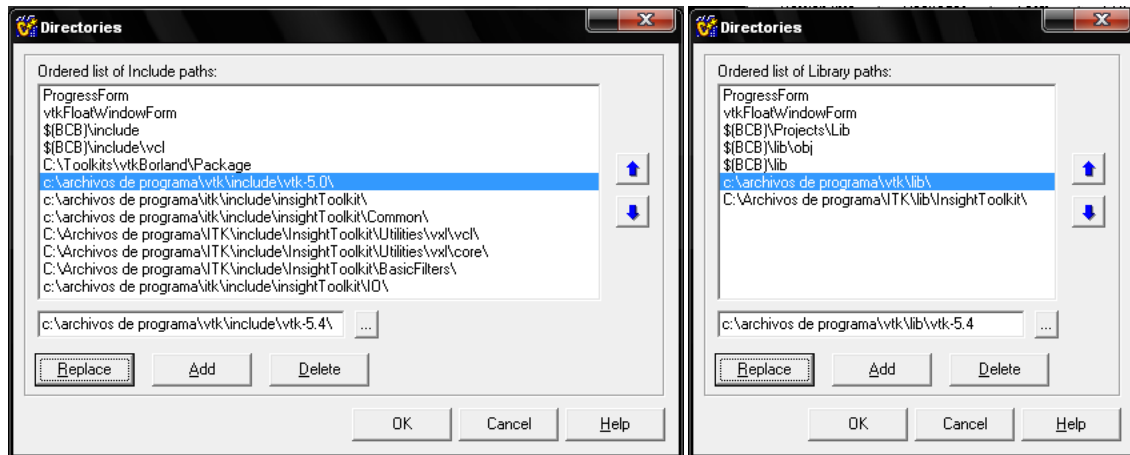


ERRORES

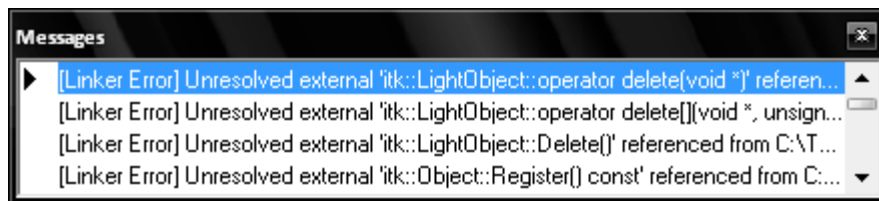
Si empiezan a surgir problemas... **!?!?!?**

Durante la ejecución de los proyectos anteriores se dieron algunos tips:

Verificar los directorios **include** y **lib**.



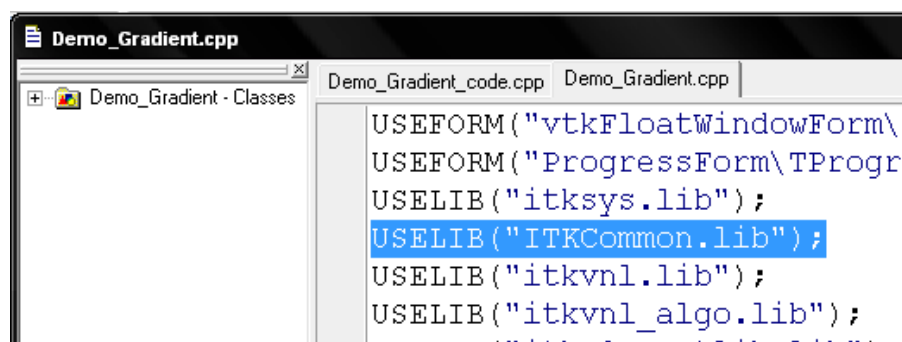
Si marca errores de LINKER



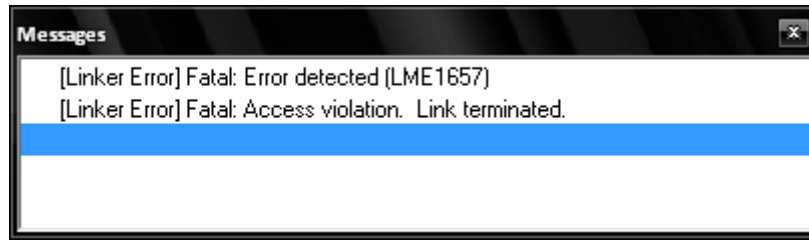
Es necesario buscar la librería que corresponde, por pertenencia de la función o por dependencia de otra librería, e incluirla, ya sea por medio de la adición al proyecto:



o manualmente:



Uno de los errores típicos y más latosos al intentar compilar proyectos conjuntos es:



Guardar TODO, cerrar Builder, borrar los archivos *.~*, *.tds, y *.obj. No siempre funciona esta acción.

Volver a abrir el proyecto, Projects->Options->Compiler opción RELEASE (botón) y activar "Disable inline expansions", esto desactiva la capacidad de *debuggear*, así que es uno de los errores a los que no te quieres enfrentar cuando estás programando un proyecto conjunto.

MAILING LISTS

Siempre existe la posibilidad de preguntar vía Internet, en particular existen dos foros y listas de correo **[Insight-Users]** y **[VTK-Users]** en las cuales puedes consultar las dudas. O bien checar archivos anteriores para ver si ya se había comentado la duda que tienes.

AL FINAL...

Sé, por experiencia propia, que al terminar de seguir este manual van a quedar más dudas que respuestas¹, solo espero que haya sido posible configurar todo y compilar los demos, y que este pequeño paso os abra el panorama de las posibilidades de investigación y generación de aplicaciones.

Si ya te sientes seguro de las instalaciones de ITK y VTK, podrías borrar los directorios [ITK_Code], [ITK_Binary], [VTK_Code] y [VTK_Binary], dado que los archivos que vas a utilizar para compilar se encuentran todos en [ITK_Install] y [VTK_Install].

Una forma de empezar a trabajar con las librerías ITK y VTK, es intentar compilar en BCB los demos que traen, y aprender sobre la práctica, aunque sin duda siempre una buena lectura ayuda.

¹ Si es la primera vez que trabajas con estas herramientas